



日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日            2 0 0 3 年   1 月 2 7 日  
Date of Application:

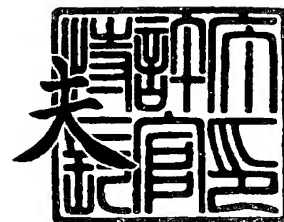
出 願 番 号            特 願 2 0 0 3 - 0 1 7 6 6 8  
Application Number:  
[ST. 10/C]:            [ J P 2 0 0 3 - 0 1 7 6 6 8 ]

出      願      人            株 式 会 社 デ ン ソ ー  
Applicant(s):

2 0 0 3 年 1 2 月   1 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康



【書類名】 特許願

【整理番号】 IP7507

【提出日】 平成15年 1月27日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/44

F02D 43/04

【発明者】

【住所又は居所】 愛知県刈谷市昭和町 1 丁目 1 番地 株式会社デンソー内

【氏名】 大井 正也

【発明者】

【住所又は居所】 愛知県刈谷市昭和町 1 丁目 1 番地 株式会社デンソー内

【氏名】 植松 義貴

【発明者】

【住所又は居所】 愛知県刈谷市昭和町 1 丁目 1 番地 株式会社デンソー内

【氏名】 岩井 明史

【特許出願人】

【識別番号】 000004260

【氏名又は名称】 株式会社デンソー

【代理人】

【識別番号】 100100022

【弁理士】

【氏名又は名称】 伊藤 洋二

【電話番号】 052-565-9911

【選任した代理人】

【識別番号】 100108198

【弁理士】

【氏名又は名称】 三浦 高広

【電話番号】 052-565-9911

## 【選任した代理人】

【識別番号】 100111578

【弁理士】

【氏名又は名称】 水野 史博

【電話番号】 052-565-9911

## 【手数料の表示】

【予納台帳番号】 038287

【納付金額】 21,000円

## 【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 コード生成装置、コード生成プログラム、シミュレーション装置、シミュレーションプログラム、モデル生成装置、およびモデル生成プログラム

【特許請求の範囲】

【請求項 1】 モデルからソースコードを生成するコード生成装置であって

、  
自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段と、

前記部分指定子によって指定された部分の取舍選択に関する選択情報を取得する選択情報取得手段と、

前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものからソースコードを生成する除去生成手段と、を備えたコード生成装置。

【請求項 2】 前記部分指定子は、指定するモデルの部分を囲む部分指定ブロックから成り、

前記除去生成手段は、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定ブロックおよび前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものからソースコードを生成することを特徴とする請求項 1 に記載のコード生成装置。

【請求項 3】 前記部分指定子は、指定する前記部分について記述される属性情報に含まれることを特徴とする請求項 1 に記載のコード生成装置。

【請求項 4】 前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報との相関関係を示す相関情報を取得する相関情報取得手段を更に備え、

前記除去生成手段は、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子、前記選択情報取得手段が取得した前記選択情報、および前記相関情報取得手段が取得した前記相関情報に基づいて、前記モデルから前記部分指定

子が指定する部分を取り除いたものからソースコードを生成することを特徴とする請求項 1 に記載のコード生成装置。

【請求項 5】 前記選択情報は、前記除去生成手段が生成するソースコードが対象とする機種の情報を含むことを特徴とする請求項 1 ないし 4 のいずれか 1 つに記載のコード生成装置。

【請求項 6】 前記選択情報は、前記除去生成手段が生成するソースコードの仕向きの情報を含むことを特徴とする請求項 1 ないし 5 のいずれか 1 つに記載のコード生成装置。

【請求項 7】 前記選択情報は、前記除去生成手段が生成するソースコードの使用目的の情報を含むことを特徴とする請求項 1 ないし 6 のいずれか 1 つに記載のコード生成装置。

【請求項 8】 コンピュータを、  
自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段、

前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段、および

前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものからソースコードを生成する除去生成手段、として機能させるためのコード生成プログラム。

【請求項 9】 自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段と、

前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段と、

前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものに記述された機能を実行する除去実行手段と、を備えたシミュレーション装置。

【請求項 1 0】 コンピュータを、

自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段、

前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段、および

前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものに記述された機能を実行する除去実行手段、として機能させるためのシミュレーションプログラム。

【請求項 1 1】 自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段と、

前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段と、

前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたモデルを生成する除去生成手段と、を備えたモデル生成装置。

【請求項 1 2】 コンピュータを、

自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段、

前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段、および

前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたモデルを生成する除去生成手段、として機能させるためのモデル生成プログラム。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、モデルによるプログラムの開発において用いられるコード生成装置

、コード生成プログラム、シミュレーション装置、シミュレーションプログラム、モデル生成装置、およびモデル生成プログラムに関するものである。

#### 【0 0 0 2】

##### 【従来の技術】

従来、例えば車両のエンジン E C U の作動のためのプログラムにおいては、開発者が直接そのプログラムのソースコードを記述せず、目的とするプログラムの機能を、より作成が簡易で視認性の良い「モデル」という形態で記述する場合がある。開発者は、このモデルに対応したプログラム開発環境がインストールされたワークステーション、パーソナルコンピュータ等を用い、そのモデルをもとに目的とするプログラムの動作の確認および試験を行い、更にそのモデルからソースコードを生成する。目的とするプログラムの動作の確認および試験の機能を担うプログラムをシミュレーションツールと呼び、モデルからソースコードを生成する機能を担うプログラムをコード生成ツールと呼ぶ。コード生成ツールやシミュレーションツールは、最初からモデル開発環境に統合されている場合もあれば、追加モジュールとしてモデル開発環境に後から組み込まれる場合もある。

#### 【0 0 0 3】

モデルに対応したプログラム開発環境としては、M a t l a b（登録商標）がある。M a t l a b（登録商標）においては、開発者は M a t l a b（登録商標）の一機能である S i m u l i n k（登録商標）を用いて、目的とするプログラムの機能を、ブロックと呼ばれる機能単位の組み合わせとして記述する。この組み合わせられたブロックの集合体がモデルである。S i m u l i n k（登録商標）のブロックとしては、例えばサイン関数を生成するブロック、ファイルからデータを読み出すブロック、入力されたデータに対する特定の四則演算を行うブロック、ブロックの組み合わせから成る上位のブロックとしてのサブシステム等がある。

#### 【0 0 0 4】

このように、プログラム開発において、開発者がモデルを生成し、コード生成ツールによってモデルからソースコードが生成され、最終的にソースコードからプログラムが生成されるという手順を取ると、当初のモデルによってプログラム

が表現されることで当該プログラムの機能の視認性が良くなる等の有益な効果を得ることができる。

#### 【0005】

また、上記したエンジン ECU のためのプログラムにおいては、V6（V 型 6 気筒）、V8（V 型 8 気筒）、I6（直列 6 気筒）等のエンジンの機種別、日本向け、欧州向け、米国向け等の仕向け別、あるいはメーカー納入用途、デバッグ用途等の目的別といったバリエーションに応じて複数の種類のプログラムを作成する場合がある。

#### 【0006】

これらのバリエーションは、互いに共通する部分を多く有するので、バリエーション毎に別個にモデルの開発を行うよりは、各バリエーションを含む 1 つのモデルを作成する方が、プログラムの開発、管理労力の低減に繋がる。

#### 【0007】

1 つのモデルからバリエーションに応じて複数の種類のソースコードを作成するためには、モデルにおいてバリエーションの違いが表現され、どのバリエーションのソースコードを生成するかを切り替えることができるようになっている必要がある。これを実現する方法としては、従来、スイッチ機能を有するブロックを用いた切り替え方法が用いられている。図 17 に、この従来の切り替え方法で表現したモデルの一部分を示す。

#### 【0008】

V6 用サブシステム 51、V8 用サブシステム 52、I6 用サブシステム 53 は、ブロックの集合体であり、それぞれが V6 エンジン用、V8 エンジン用、I6 エンジン用のバリエーションに対応した処理が記述されている。これらのサブシステムは、それぞれが有するブロックによって規定された所定の演算結果を出力するようになっている。入力端子 55 は、選択ブロック 54 に対して設定可能な値としての選択信号 a を入力する端子である。

#### 【0009】

選択ブロック 54 は、入力端子 55 からの入力値に基づいて、V6 用サブシステム 51、V8 用サブシステム 52、I6 用サブシステム 53 から入力される演



算結果のいずれか 1 つを出力する機能を有するブロックである。

#### 【0010】

このような部分を有するモデルが、コード生成ツールによってソースコードに変換された場合、そのソースコードのうち図 17 に示した部分に対応する部分は、例えば図 18 の様になる。

#### 【0011】

この図のソースコード中の `Switch` 文は、中括弧 `{ }` で囲まれた領域において、引数 `a` の値が 1、2、または 3 の場合、それぞれ `case 1:`、`case 2:`、`case 3:` と直後の `break` 文との間にある文を実行することを規定する文である。`Switch` 文が選択ブロック 54 に対応し、引数 `a` が選択信号 `a` に対応し、`case 1:` から直後の `break` 文までの間のコードが V6 用サブシステム 51 に対応し、`case 2:` から直後の `break` 文までの間のコードが V8 用サブシステム 52 に対応し、`case 3:` から直後の `break` 文までの間のコードが I6 用サブシステム 53 に対応する。

#### 【0012】

##### 【発明が解決しようとする課題】

このように生成されたソースコードは、例えば設定された選択信号 `a` の値が 1 であれば V6 用サブシステム 51 に対応するソースコードの部分が実行されるようになり、V8、I6 に対応するソースコードの部分は実行されることがない。したがって、生成されたコードは選択信号 `a` によってそれぞれのバリエーションに対応することができるが、使用することのないソースコードの部分まで形式的に含んでいるので、このソースコードがコンパイルされると、無駄なソースコード部分を含むプログラムが生成され、プログラムサイズが増大してしまう。この結果、プログラムを格納するメモリの容量が圧迫されてしまう。

#### 【0013】

本発明は上記点に鑑みて、複数のバリエーションに対応するモデルからソースコードが生成されるプログラム開発環境において、生成されるソースコードから当初のモデルの不要な部分に対応するソースコードが除かれているようにすることを目的とする。

**【0014】**

またプログラム開発環境においては、モデルからソースコードが生成されるまでの間の段階として中間コードが生成される場合もあることに鑑みて、複数のバリエーションに対応するモデルからさらにモデルが生成されるプログラム開発環境において、生成されるモデルから当該モデルの不要な部分に対応するモデルが除かれているようにすることも目的とする。

**【0015】**

また、上記のようなプログラム開発環境において、モデルからオブジェクトコードを生成する前に、あらかじめ当該オブジェクトコードの動作を確認するために、当該モデルを直接実行するシミュレーションプログラムまたは装置が提供されることがある。このようなシミュレーションプログラム等は、モデル生成されるオブジェクトコードと同等の動作を実現する必要があることに鑑み、複数のバリエーションに対応するモデルを実行して動作を確認するプログラム開発環境において、実行されるモデルを、当初のモデルの不要な部分に対応する機能を除いた形で実行することも目的とする。

**【0016】****【課題を解決するための手段】**

上記した課題を解決するための請求項1に記載の発明は、モデルからソースコードを生成するコード生成装置であって、自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段と、前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段と、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものからソースコードを生成する除去生成手段と、を備えたコード生成装置である。

**【0017】**

これによって、除去生成手段が、モデル取得手段が取得したモデルに含まれる部分指定子および選択情報取得手段が取得した選択情報に基づいて、モデルから部分指定子が指定する部分を取り除いたものからソースコードを生成するので、

部分指定子で指定された、不要となり得るモデルの部分は、選択情報によっては生成されるソースコードから取り除かれる。したがって、複数のバリエーションに対応するモデルからソースコードが生成されるプログラム開発環境において、生成されるソースコードから当該モデルの不要な部分に対応するソースコードが除かれているようにすることが可能となる。

#### 【0018】

なお、部分指定子は1つのモデル中に複数含まれていても構わない。また、除去生成手段が「モデルから部分指定子が指定する部分を取り除いたもの」とは、モデルに含まれる部分指定子のうちの1つまたは複数が指定する部分を取り除いたもの、およびモデルに含まれる「全ての」部分指定子が指定する部分を取り除いたものを含む概念である。

#### 【0019】

また、請求項2に記載の発明は、請求項1に記載のコード生成装置において、前記部分指定子は、指定するモデルの部分を囲む部分指定ブロックから成り、前記除去生成手段は、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定ブロックおよび前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものからソースコードを生成することを特徴とする。

#### 【0020】

また、請求項3に記載の発明は、請求項1に記載のコード生成装置において、前記部分指定子は、指定する前記部分について記述される属性情報に含まれることを特徴とする。

#### 【0021】

また、請求項4に記載の発明は、請求項1に記載のコード生成装置において、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報との相関関係を示す相関情報を取得する相関情報取得手段を更に備え、前記除去生成手段は、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子、前記選択情報取得手段が取得した前記選択情報、および前記相関情報取得手段が取得した前記相関情報に基づいて

、前記モデルから前記部分指定子が指定する部分を取り除いたものからソースコードを生成することを特徴とする。

【0 0 2 2】

また、請求項 5 に記載の発明は、請求項 1 ないし 4 のいずれか 1 つに記載のコード生成装置において、前記選択情報は、前記除去生成手段が生成するソースコードが対象とする機種の情報を含むことを特徴とする。

【0 0 2 3】

また、請求項 6 に記載の発明は、請求項 1 ないし 5 のいずれか 1 つに記載のコード生成装置において、前記選択情報は、前記除去生成手段が生成するソースコードの仕向けの情報を含むことを特徴とする。

【0 0 2 4】

また、請求項 7 に記載の発明は、請求項 1 ないし 6 のいずれか 1 つに記載のコード生成装置において、前記選択情報は、前記除去生成手段が生成するソースコードの使用目的の情報を含むことを特徴とする。

【0 0 2 5】

また、請求項 8 に記載の発明は、請求項 1 に記載の発明と同じ機能をプログラムとして実現するものである。

【0 0 2 6】

また、請求項 9 に記載の発明は、自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段と、前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段と、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたものに記述された機能を実行する除去実行手段と、を備えたシミュレーション装置である。

【0 0 2 7】

これによって、除去実行手段が、モデル取得手段が取得したモデルに含まれる部分指定子および選択情報取得手段が取得した選択情報に基づいて、モデルから部分指定子が指定する部分を取り除いたものに記述された機能を実行するので、

部分指定子で指定された、不要となり得るモデルの部分は、選択情報によっては生成されるモデルから取り除かれる。

【 0 0 2 8 】

したがって、複数のバリエーションに対応するモデルを実行して動作を確認するプログラム開発環境において、実行されるモデルから当該モデルの不要な部分に対応するソースコードが除かれているようにすることが可能となる。

【 0 0 2 9 】

また、請求項 1 0 に記載の発明は、請求項 9 に記載の発明と同じ機能をプログラムとして実現するものである。

【 0 0 3 0 】

また、請求項 1 1 に記載の発明は、自己の特定の部分を指定する部分指定子を含むモデルを取得するモデル取得手段と、前記部分指定子によって指定された部分の取捨選択に関する選択情報を取得する選択情報取得手段と、前記モデル取得手段が取得した前記モデルに含まれる前記部分指定子および前記選択情報取得手段が取得した前記選択情報に基づいて、前記モデルから前記部分指定子が指定する部分を取り除いたモデルを生成する除去生成手段と、を備えたモデル生成装置である。

【 0 0 3 1 】

これによって、除去生成手段が、モデル取得手段が取得したモデルに含まれる部分指定子および選択情報取得手段が取得した選択情報に基づいて、モデルから部分指定子が指定する部分を取り除いたモデルを生成するので、部分指定子で指定された、不要となり得るモデルの部分は、選択情報によっては生成されるモデルから取り除かれる。したがって、複数のバリエーションに対応するモデルを実行して動作を確認するプログラム開発環境において、実行されるモデルを、当該モデルの不要な部分に対応する機能を除いた形で実行することが可能となる。

【 0 0 3 2 】

また、請求項 1 2 に記載の発明は、請求項 1 1 に記載の発明と同じ機能をプログラムとして実現するものである。

【 0 0 3 3 】

**【発明の実施の形態】****(第1実施形態)**

図1に、本発明の第1実施形態に係る、コード生成装置としてのパーソナルコンピュータ1の構成を示す。パーソナルコンピュータ1は、ディスプレイ11、入力装置12、RAM13、ROM14、HDD15、およびCPU16から構成される。

**【0034】**

ディスプレイ11は、CPU16から入力された映像信号を、ユーザに対して映像として表示する。

**【0035】**

入力装置12は、キーボード、マウス等から構成され、ユーザが操作することにより、その操作に応じた信号をCPU16に出力する。

**【0036】**

CPU16は、パーソナルコンピュータ1に電源が投入されることによって起動すると、ROM（リードオンリーメモリ）14から所定のブートプログラムを読み出して実行し、このブートプログラムに規定されるMS-Windows（登録商標）等のオペレーティングシステム（以下OSと記す）その他のプログラムをHDD（ハードディスクドライブ）15から読み出して実行することにより、起動処理を行う。起動処理以後電源が遮断されるまで、CPU16は、入力装置12からの信号、OSによって予め定められたスケジュール等に基づいて、HDD15に記録されている各種プログラムを当該OS上のプロセスとして実行する。また、上記した起動処理およびプロセスにおいて、CPU16は必要に応じて入力装置12から信号の入力を受け付け、またディスプレイ11に映像信号を出力し、またRAM13、HDD15に対してデータの読み出し／書き込みの制御を行う。

**【0037】**

図2に、HDD15に保存され、OS上のプロセスとして実行されるプログラムの1つであるコード生成ツール2の構成および作動の概念図を示す。コード生成ツール2は、OSの仕様に従ったユーザの入力装置12の操作によって起動し

、その後ユーザの入力装置 12 の操作に基づいて、モデルからソースコードを生成するプログラムである。

#### 【0038】

ソースコードとは、例えば C++ 等のプログラミング言語の仕様に則ってプログラム開発者が作成するプログラムの一表現形態である。ソースコードはコンパイラ、リンカ等に入力されることで、CPU 等が直接実行できるオブジェクトコードに変換される。なお、オブジェクトコードもプログラムの一表現形態である。

#### 【0039】

モデルとは、ソースコードよりも記述が簡易になり、かつ人による可読性が高くなる目的で定められたモデル言語仕様にに基づいて作成されるプログラムの一表現形態である。モデルとしては、例えば Simulink (登録商標) によって作成される Simulink モデルがある。Simulink (登録商標) モデルが、Matlab (登録商標) 上で動作する Real Time Workshop (登録商標: 以下 RTW と記す) に入力されることにより、この RTW が当該 Simulink (登録商標) モデルに対応するソースコードを生成する。

#### 【0040】

Simulink (登録商標) モデルは、ブロックと呼ばれる機能単位の組み合わせからなる集合体として作成される。組み合わせとは、機能単位間の入出力の繋がりをいう。ブロックの具体例としては、例えば入力される 2 つの数値データを加算して出力する加算ブロック等がある。また、複数のブロックの集合体をサブシステムというブロックの一種として定義することもできる。すなわち、ブロックの中にブロックが含まれるような入れ子構造のブロックの作成も許される。また、Simulink (登録商標) モデル中の各ブロックは、それぞれ属性情報を有することができる。属性情報としては、例えば当該ブロックの名称、入出力するデータの値域定義等がある。

#### 【0041】

本実施形態におけるモデルも、Simulink (登録商標) モデルと同様にブロックから成り、ブロックの入れ子構造が許され、各ブロックが属性情報を有

することができる。

【0042】

コード生成ツール 2 は、その機能面から生成モデル抽出エンジン 2 1、コード生成エンジン 2 2、生成ルール 2 3 に分けることができる。

【0043】

生成モデル抽出エンジン 2 1 は、入力されるモデル 3 および選択情報 4 に基づいて中間モデルを生成する。この中間モデルは、モデルの一種である。

【0044】

コード生成エンジン 2 2 は、生成モデル抽出エンジン 2 1 によって生成された中間モデルからソースコードを生成する。この中間モデルからソースコードを生成するための規則を定めた情報が生成ルール 2 3 である。生成ルールはコード生成ツール 2 の一部として存在していてもよいし、あるいはコード生成ツール 2 の外部のファイルとして HDD 1 5 に保存されていてもよい。なお、生成モデル抽出エンジン 2 1 とコード生成エンジン 2 2 との区別は、あくまでもコード生成ツール 2 の機能面から見た区別であって、必ずしもプログラムとしては分離している必要はない。実際、本実施形態においては、生成モデル抽出エンジン 2 1 とコード生成エンジン 2 2 とは一体のプログラムとして実現される。

【0045】

生成されたソースコードは、HDD 1 5 に記録されているコンパイラ、リンカ 5 を実行することによってオブジェクトコードに変換される。

【0046】

図 3 に、コード生成ツール 2 に入力されるモデル 3 の一例を示す。このモデルは、車両のエンジン ECU に搭載されるプログラムのためのモデルの一部であり、V6 スタート 3 1、V6 用サブシステム 3 2、V6 エンド 3 3、V8 スタート 3 4、V8 用サブシステム 3 5、V8 エンド 3 6、I6 スタート 3 7、I6 用サブシステム 3 8、および I6 エンド 3 9 から構成される。

【0047】

V6 スタート 3 1、V6 エンド 3 3、V8 スタート 3 4、V8 エンド 3 6、I6 スタート 3 7、および I6 エンド 3 9 は部分指定ブロックである。部分指定ブ



ロックは上述したブロックの一種であり、本発明における部分指定子に相当する。さらに部分指定ブロックは、V6 スタート 31、V8 スタート 34、I6 スタート 37 のスタートブロック、および V6 エンド 33、V8 エンド 36、I6 エンド 39 のエンドブロックに分類される。1 種類のスタートブロックには必ず対応する同種のエンドブロックが存在する。図 3 においては、V6 スタート 31 と V6 エンド 33、V8 スタート 34 と V8 エンド 36、および I6 スタート 37 と I6 エンド 39 がそれぞれ対応するブロックである。

#### 【0048】

V6 用サブシステム 32、V8 用サブシステム 35、および I6 用サブシステム 38 は、ブロックの集合体を有するブロック、すなわち入れ子構造のブロックである。それぞれのサブシステムは、V6 エンジン、V8 エンジン、I6 エンジンといった、対象とする機種の制御に特有な機能を記述するブロックから構成されている。

#### 【0049】

なお、上記したそれぞれのブロックは、V6 スタート、I6 サブシステム等の自己の名称を属性情報として有している。また、スタートブロックは自己がスタートブロックである旨の情報を属性情報に有し、エンドブロックは自己がエンドブロックである旨の情報を属性情報に有している。

#### 【0050】

部分指定ブロックは、互いに対応する 1 対のスタートブロックとエンドブロックとの組としてモデル 3 中に現れる。そして、ブロック中のスタートブロックと、それに対応するエンドブロックとに前後を囲まれるブロック、サブシステム等は、当該部分指定ブロックによって、V6 用、V8 用、I6 用等のバリエーションに対応する部分であると指定されることになる。

#### 【0051】

図 4 に、各サブシステムにおいてブロックの記述が異なる例として、入力されるエンジンの各気筒の状態を加算することでエンジン全体の状態を出力する機能を表現するブロックを示す。図 4 (a) が V6 および I6 のサブシステム用のブロックであり、図 4 (b) が V8 のサブシステム用のブロックである。V6、I

6のエンジンは気筒数が6であるので、6本の入力端子を有するブロックが用いられる。また、V8のエンジンは気筒数が8であるので、8本の入力端子を有するブロックが用いられる。

#### 【0052】

図5に、コード生成ツール2において、ユーザが所定の操作をすることによって開始される、モデルからソースコードを生成する処理を示す。所定の操作とは、ユーザがコード生成ツール2に入力するモデル3、選択情報4を指定するために入力装置12を操作し、さらに図5の処理を開始するよう要求するために入力装置12を操作することをいう。なお、本実施形態においては、選択情報4は特定のスタートブロックの名称である。また、選択情報4は、HDD15中の特定のファイル中に予め設定されていてもよいし、ユーザが選択情報4の指定時に直接入力して設定してもよい。以下、図5の処理について説明する。

#### 【0053】

まずステップ510で、ユーザによって指定されたモデル3を読み込む。具体的には、モデル3として保存されているHDD15中のファイルを読み出してRAM13に書き込む。

#### 【0054】

次にステップ520で、ユーザによって指定された選択情報4を読み込む。具体的には、選択情報4として保存されているHDD15中のファイルを読み出してRAM13に書き込む。あるいは、入力装置12から直接設定された情報を選択情報4としてRAM13に書き込む。

#### 【0055】

次にステップ530では、モデル3中のスタートブロックの検索を行う。具体的には、RAM13に書き込んだモデル3に含まれる全てのブロックについて、それがスタートブロックであるか否かを判定し、スタートブロックであると判定したもの全てについて、そのブロックの属性情報、モデル3内の位置等の情報をスタートブロックリストとしてRAM13に書き込む。

#### 【0056】

次にステップ535で、スタートブロックリストの先頭にあるスタートブロッ

クの情報を読み出し、その後読み出したスタートブロックの情報をスタートブロックリストから削除する。先頭のスタートブロックの情報が削除されることで、次のスタートブロックの情報が先頭のスタートブロックの情報となる。

#### 【 0 0 5 7 】

次にステップ 5 4 0 で、当該スタートブロックの名称が R A M 1 3 に記録された選択情報 4 と一致しているか否かを判定する。一致していると判定すると、処理はステップ 5 5 5 に進む。一致していないと判定すると、処理はステップ 5 5 0 に進む。

#### 【 0 0 5 8 】

ステップ 5 5 0 では、当該スタートブロックおよびそれと対になるエンドブロックに囲まれた部分、すなわち当該部分指定子によって指定される部分を削除する。具体的には、当該スタートブロックの位置から、モデル 3 における後段方向へ向けて、当該スタートブロックと対になるエンドブロックを検索する。そして R A M 1 3 中のモデル 3 を、見つかったエンドブロックから当該スタートブロックまでのブロックが削除されたモデル 3 に書き換える。そして処理はステップ 5 5 5 に進む。

#### 【 0 0 5 9 】

ステップ 5 5 5 では、当該スタートブロックおよびこれと対になるエンドブロックを、ステップ 5 5 0 で書き換えられたモデル 3 から削除する。すなわち、R A M 1 3 中のモデル 3 を、そのモデル 3 から当該スタートブロックおよびエンドブロックが削除されたモデル 3 に書き換える。

#### 【 0 0 6 0 】

次にステップ 5 6 0 では、当該スタートブロックが最後のスタートブロックであるか否かを判定する。具体的には、スタートブロックリストにスタートブロックの情報が無ければ肯定であると判定し、あれば否定であると判定する。否定と判定すれば、処理はステップ 5 3 5 に戻る。肯定の場合、処理はステップ 5 7 0 に進み、R A M 1 3 に記録されているモデル 3 から、生成ルール 2 3 に従ってソースコードを生成し、生成されたソースコードを H D D 1 5 中にファイルとして書き込む。そして処理は終了する。

**【0061】**

このような作動によって、例えば図3で示されたブロックを含むモデル3と、「V6」を示す選択情報4を入力してコード生成ツール2を作動させると、コード生成ツール2のステップ510～560の処理によって、図6に示すような中間モデルが生成される。そしてステップ570によってこの中間モデルに対応するソースコードが生成ルール23に従って生成される。したがって、この生成されたソースコードにはV8用、I6用に特化されたコードは含まれることはない。

**【0062】**

なお、ステップ510～560の処理は、図2に示した生成モデル抽出エンジン21の機能を実現し、ステップ570の処理は、コード生成エンジン22の機能を実現する。

**【0063】**

以上のような構成および作動のパーソナルコンピュータ1、コード生成ツール2の効果について以下説明する。

**【0064】**

V6、V8、I6等、複数のバリエーションに対応するモデルを作成し、ソースコードの作成時に個々のバリエーションにのみ向けられたソースコードを作成したい場合、それぞれのバリエーションに対応する部分を、そのバリエーションに固有の名称を持つ部分指定ブロックで囲んだモデル3を作成する。そして、ソースコード作成時に、選択情報4として特定のバリエーションの名称をファイルまたはユーザによる操作によってコード生成ツール2に入力し、また当該モデル3をコード生成ツール2に入力する。

**【0065】**

これによって、コード生成ツール2は選択情報4に示された名称以外の名称の部分指定ブロックに囲まれたサブシステム等をモデル3から削除した上でソースコードを生成する。したがって、複数のバリエーションに対応するモデルからソースコードが生成される開発環境において、当該ソースコードから不要なバリエーションに対応するソースコードが除かれていることになる。ひいては、このソ

ースコードからコンパイル、リンクによって生成されるプログラムサイズの低減、およびこのプログラムを記録するメモリの容量の圧迫の緩和が実現される。

#### 【0066】

また、複数のバリエーションに対応するモデルからさらにモデルが生成されるプログラム開発環境において、生成されるモデルから当初のモデルの不要な部分に対応するモデルが除かれているようにすることも可能となる。

#### 【0067】

##### (第2実施形態)

以下、本発明の第2実施形態について説明する。本実施形態も、図1および図2で示したようなハードウェア構成およびソフトウェア構成から成っている。なお、本実施形態において第1実施形態と同一の部分があれば、その説明は省略または簡略化する。

#### 【0068】

本実施形態が第1実施形態と異なるのは、第1実施形態の部分指定子が部分指定ブロックであったことに対し、本実施形態の部分指定子がサブシステムの属性情報に含まれることである。

#### 【0069】

図7に、本実施形態においてコード生成ツール2に入力されるモデル3の一部を例示する。このモデル3は、Aサブシステム71、Bサブシステム72、Cサブシステム73、Dサブシステム74、およびEサブシステム75を有している。また、上記したサブシステムは、それぞれ属性情報71a、72a、73a、74a、75aを有している。各属性情報は、部分指定子としての選択情報の項目を有し、その項目の値として、属性情報71aにはV6が、属性情報72aにはV8が、属性情報73aにはI6が、属性情報74aにはV6およびV8が、属性情報75aにはI6が、設定されている。

#### 【0070】

図8に、本実施形態のコード生成ツール2において、ユーザが所定の操作をすることによって開始される、モデルからソースコードを生成する処理を示す。

#### 【0071】

ステップ 810、820 の処理は図 3 に示したステップ 510、520 の処理と同等である。ただし、本実施形態においては、選択情報 4 は、V6、V8、I6 等の選択情報の値である。

#### 【0072】

次にステップ 825 では、モデル 3 中のサブシステムの検索を行う。具体的には、RAM13 に書き込んだモデル 3 に含まれる全てのブロックについて、サブシステムであるか否かを判定し、サブシステムであると判定したもの全てについて、そのサブシステムの属性情報、モデル 3 内の位置等の情報をサブシステムリストとして RAM13 に書き込む。

#### 【0073】

次にステップ 830 で、サブシステムリストの先頭にあるサブシステムの属性情報等の情報を読み出し、その後当該サブシステムの情報をサブシステムリストから削除する。そしてステップ 840 で、読み出した属性情報中の使用バリエーションの項目に、選択情報 4 と一致するものがあるか否かを判定する。一致するものが無ければ処理はステップ 850 に進み、RAM13 中のモデル 3 を、モデル 3 から当該サブシステムが削除されたものに書き換え、その後ステップ 860 に進む。ステップ 840 で一致するものがあれば、処理は直接ステップ 860 に進む。

#### 【0074】

ステップ 860 では、全てのサブシステムの属性情報について読み込みが完了したか否かを判定する。具体的には、サブシステムリストにサブシステムの情報が無ければ肯定であると判定し、あれば否定であると判定する。否定と判定すれば、処理はステップ 830 に戻る。肯定の場合、処理はステップ 870 に進み、RAM13 に記録されているモデル 3 から、生成ルール 23 に従ってソースコードを生成し、生成されたソースコードを HDD15 中にファイルとして書き込む。そして処理は終了する。

#### 【0075】

このような作動によって、例えば図 7 で示されたブロックを含むモデル 3 と、「V6」を示す選択情報 4 を入力してコード生成ツール 2 を作動させると、コー

ド生成ツール 2 のステップ 810～860 の処理によって、図 9 に示すような中間モデルが生成される。そしてステップ 870 によってこの中間モデルに対応するソースコードが生成される。したがって、この生成されたソースコードには V8 用、I6 用に特化されたコードは含まれることはない。

#### 【0076】

なお、ステップ 810～860 の処理は、図 2 に示した生成モデル抽出エンジン 21 の機能を実現し、ステップ 870 の処理は、コード生成エンジン 22 の機能を実現する。

#### 【0077】

以上のような構成および作動のパーソナルコンピュータ 1、コード生成ツール 2 の効果について以下説明する。

#### 【0078】

V6、V8、I6 等、複数のバリエーションに対応するモデルを作成し、ソースコードの作成時に個々のバリエーションにのみ向けられたソースコードを作成したい場合、それぞれのバリエーションに対応する部分をサブシステムとし、そのサブシステムの属性情報中に、使用バリエーションの項目を設け、その項目の値を、当該サブシステムが対応するバリエーションとする。そして、ソースコード作成時に、選択情報 4 として特定のバリエーションの名称をファイルまたはユーザによる操作によってコード生成ツール 2 に入力し、また当該モデル 3 をコード生成ツール 2 に入力する。なお、使用バリエーションの項目の値としては、V6、V8、I6 のいずれか 1 つであってもよいし、あるいはそれらのうちの任意の 2 つ、あるいは 3 つ全部を含む値であってもよい。例えば、V6、I6 に共通に用いられるサブシステムは、使用バリエーションの項目として V6、I6 の 2 つの値を含むようにする。また、全バリエーションで共通して用いられるサブシステムについては、使用バリエーションの項目として V6、V8、I8 の全ての値を含むようにすればよい。このように、サブシステムの属性情報中の使用バリエーションの項目値が、部分指定子に対応する。

#### 【0079】

これによって、コード生成ツール 2 は選択情報 4 に示された名称を使用バリエ

ーションの項目の値に含まないサブシステムをモデル 3 から削除した上で、ソースコードを生成する。したがって、複数のバリエーションに対応するモデルからソースコードが生成される開発環境において、当該ソースコードから不要なバリエーションに対応するソースコードが除かれていることになる。ひいては、このソースコードからコンパイル、リンクによって生成されるプログラムサイズの低減、およびこのプログラムを記録するメモリの容量の圧迫の緩和が実現される。

#### 【0080】

また、複数のバリエーションに対応するモデルからさらにモデルが生成されるプログラム開発環境において、生成されるモデルから当初のモデルの不要な部分に対応するモデルが除かれているようにすることも可能となる。

#### 【0081】

なお、別の例として、自己の属性情報中に使用バリエーションの項目を有さないサブシステム、または使用バリエーションの項目の値が V 6、V 8、I 6 のいずれをも含んでいないサブシステムについては、使用バリエーションの項目として V 6、V 8、I 8 の全ての値を含むとみなしてステップ 840 の処理を行ってもよい。これによって、全てのバリエーションに共通に用いられるサブシステムについては、その属性情報中に能動的にその旨を記述する必要がなくなる。

#### 【0082】

(第 3 実施形態)

以下、本発明の第 3 実施形態について説明する。なお、本実施形態において第 2 実施形態と同一の部分があれば、その説明は省略または簡略化する。

#### 【0083】

図 10 に、本実施形態におけるコード生成ツール 2 の構成、作動を概略的に示す。

#### 【0084】

本実施形態が第 2 実施形態と異なるのは、第 1 に、第 2 実施形態の部分指定子が属性情報中の使用バリエーションの項目値であったことに対し、本実施形態の部分指定子がサブシステムの名称となることである。そして第 2 に、選択情報 4 と部分指定子との相関関係を示す相関情報として切替マトリクス 6 が用いられる



ことである。

#### 【0085】

図11に、本実施形態においてコード生成ツール2に入力されるモデル3の一部を例示する。このモデル3は、Aサブシステム81、Bサブシステム82、Cサブシステム83、Dサブシステム84、およびEサブシステム85を有している。それぞれのサブシステムの名称の情報は、それぞれのサブシステムの属性情報に含まれている。

#### 【0086】

図12に、切替マトリクス6の一例を示す。切替マトリクス6は、モデル3に含まれる全サブシステムを行項目として有し、V6、V8、I6といったバリエーションを列項目として有する行列を示す情報である、切替マトリクス6の各セルには1つのフラグが割り当てられている。図12においては、フラグがセットされているセルを丸印で示し、フラグがセットされていない、すなわちリセットされているセルを空白で示している。各フラグのセット、リセットは、あらかじめユーザが設定できる。このように、切替マトリクス6は行項目のバリエーションと列項目のサブシステムとの相関関係を表している。

#### 【0087】

図13に、本実施形態のコード生成ツール2において、ユーザが所定の操作をすることによって開始される、モデルからソースコードを生成する処理を示す。

#### 【0088】

ステップ910、920の処理は図8に示したステップ810、820の処理と同等である。

#### 【0089】

そしてステップ923では、ユーザによって指定された切替マトリクス6を読み込む。具体的には、切替マトリクス6として保存されているHDD15中のファイルを読み出す。更に読み出した切替マトリクス6の、ステップ920で読み出した選択情報4が示すバリエーションに相当する列項目中で、フラグがセットされているセルを検索し、該当する各セルが存在する行のサブシステムのリストを選択リストとしてRAM13に記録する。

**【0090】**

次にステップ925では、図8のステップ825と同等の処理でモデル3中のサブシステムの検索を行う。

**【0091】**

次にステップ930で、サブシステムリストの先頭にあるサブシステムの名称の情報を読み出し、その後当該サブシステムの情報をサブシステムリストから削除する。そしてステップ940で、読み出した属性情報中のサブシステムの名称が、選択リストと一致するか否かを判定する。一致しなければ処理はステップ950に進み、RAM13中のモデル3を、モデル3から当該サブシステムが削除されたものに書き換え、その後ステップ960に進む。ステップ940で一致すれば、処理は直接ステップ960に進む。

**【0092】**

ステップ960では、全てのサブシステムの名称について読み込みが完了したか否かを判定する。具体的な判定処理は図8のステップ860と同等である。否定と判定すれば、処理はステップ930に戻る。肯定の場合、処理はステップ970に進み、RAM13に記録されているモデル3から、生成ルール23に従ってソースコードを生成し、生成されたソースコードをHDD15中にファイルとして書き込む。そして処理は終了する。

**【0093】**

このような作動によって、例えば図11で示されたブロックを含むモデル3と、「V6」を示す選択情報4と、図12のようにセルをセットした切替マトリクス6とを入力してコード生成ツール2を作動させると、コード生成ツール2のステップ910～960の処理によって、図14に示すような中間モデルが生成される。そしてステップ970によってこの中間モデルに対応するソースコードが生成される。したがって、この生成されたソースコードにはV8用、I6用に特化されたコードは含まれることはない。

**【0094】**

なお、ステップ910～960の処理は、図10に示した生成モデル抽出エンジン21の機能を実現し、ステップ970の処理は、コード生成エンジン22の

機能を実現する。

【0095】

以上のような構成および作動のパーソナルコンピュータ 1、コード生成ツール 2 の効果について以下説明する。

【0096】

V 6、V 8、I 6 等、複数のバリエーションに対応するモデルを作成し、ソースコードの作成時に個々のバリエーションにのみ向けられたソースコードを作成したい場合、それぞれのバリエーションに対応する部分をサブシステムとし、それらのサブシステムの名称とバリエーションとの相関関係を示す切替マトリクス 6 を作成する。そして、ソースコード作成時に、選択情報 4 として特定のバリエーションの名称をファイルまたはユーザによる操作によってコード生成ツール 2 に入力し、また作成された切替マトリクス 6 をコード生成ツール 2 に入力し、また当該モデル 3 をコード生成ツール 2 に入力する。

【0097】

これによって、コード生成ツール 2 は選択情報 4 に示されたバリエーションおよび切替マトリクス 6 のセットされたフラグによって相関づけられたサブシステム以外のサブシステムをモデル 3 から削除した上で、ソースコードを生成する。したがって、複数のバリエーションに対応するモデルからソースコードが生成される開発環境において、当該ソースコードから不要なバリエーションに対応するソースコードが除かれていることになる。ひいては、このソースコードからコンパイル、リンクによって生成されるプログラムサイズの低減、およびこのプログラムを記録するメモリの容量の圧迫の緩和が実現される。

【0098】

また、複数のバリエーションに対応するモデルからさらにモデルが生成されるプログラム開発環境において、生成されるモデルから当初のモデルの不要な部分に対応するモデルが除かれているようにすることも可能となる。

【0099】

また、第 1 および第 2 実施形態と異なり、モデル 3 の各サブシステムは自己の名称のみを有していればよいので、モデル 3 がバリエーションの切り替えのため

の情報を特別に有する必要がなく、切替マトリクス 6 と選択情報 4 のみで切り替えのための情報を一元管理することができる。

#### 【0100】

##### (第 4 実施形態)

図 15 に、本発明の第 4 実施形態に係るシミュレーションツール 7 の構成、作動を概略的に示す。

#### 【0101】

シミュレーションツール 7 は、入力されたモデル 3 からソースコードを生成しないまま、パーソナルコンピュータ 1 上でそのモデル 3 によって表現されるプログラムを実行するプログラムである。これは、モデル 3 からコード生成ツール 2 によってソースコードを生成し、そのソースコードから作成されたオブジェクトコードをエンジン ECU 等を実装する前に、モデル 3 に基づくプログラムの動作の試験、確認を行うためのものである。

#### 【0102】

シミュレーションツール 7 は、その機能面からシミュレーションモデル抽出エンジン 24 およびシミュレーションエンジン 25 に分けることができる。

#### 【0103】

シミュレーションモデル抽出エンジン 24 は、第 3 実施形態における生成モデル抽出エンジン 21 と同等の機能を有する。具体的には、モデル 3、選択情報 4、切替マトリクス 6 の入力に対して、図 13 のステップ 910～960 に示したような処理を行い、中間モデルを生成する。

#### 【0104】

シミュレーションエンジン 25 は、シミュレーションモデル抽出エンジン 24 が生成した中間モデルによって表現されたプログラムの機能を、パーソナルコンピュータ 1 上で実行する。具体的には、シミュレーションエンジン 25 は、当該中間モデルを読み込み、その中間モデルをソースコードに変換することなく、それに表現された機能を実行する。そしてその実行の際、ユーザの選択による任意のブロックの入力データ、出力データ、およびそれらの相関関係等の情報をディスプレイ 11 あるいはプリンタ等に出力する。データの相関関係の情報としては

、例えばアクセル開度と燃料噴射量との関係を示すグラフ等がある。

【0105】

なお、シミュレーションモデル抽出エンジン24とシミュレーションエンジン25との区別は、あくまでもコード生成ツール2の機能面から見た区別であって、必ずしもプログラムとしては分離している必要はなく、一体不可分のプログラムとして実現されていてもよい。

【0106】

モデル3、選択情報4、切替マトリクス6等については、第3実施形態と同様である。ただし、シミュレーションエンジン25が当該モデルとそのモデルが実装されるエンジンECU等のハードウェアとの間の信号の入出力を擬似的に実現するために、モデル3にはハードウェアからの入力信号を代替するブロックが付加されている。この付加されたブロックは、第3実施形態のコード生成ツール2に当該モデルを入力する際には取り除く必要がある。

【0107】

このようなシミュレーションツール7によって、複数のバリエーションに対応するモデルを実行して動作を確認するプログラム開発環境において、実行されるモデルを、当初のモデルの不要な部分に対応する機能を除いた形で実行することが可能となる。

【0108】

このような場合、モデル3は、実際にはエンジンECU等の特定のハードウェア上のみで実行されるプログラムとして開発されとしても、プログラムの動作の試験、確認については、当該ハードウェアにプログラムを実装する前に行うことができるので、開発の利便性が向上する。

【0109】

(他の実施形態)

上記した各実施形態においては、V6、V8、I6といった、生成されるプログラムが対象とする機種についてのバリエーションを例示し、選択情報4はこの対象とする機種の情報を含むようになっていた。しかし、必ずしも選択情報4が含むバリエーションは生成されるプログラムが対象とする機種についてのもので

ある必要はなく、例えば日本向け、米国向け、ヨーロッパ向けといった仕向けについてのものであってもよいし、または試験用、量産用といったプログラムの使用目的についてのものであってもよい。仕向けによってプログラムの機能が異なる例としては、例えば各国内法のエンジンに対する規制が異なるため、それぞれの法規制に対応する部分が異なっている場合がある。

#### 【0110】

図16に、仕向けによって異なるブロックの構成例を示す。(a)は米国用に作成されたサブシステム中の一部のブロック構成であり、(b)は米国以外用に作成されたサブシステム中の、(a)に対応する部分である。ブロック91は、米国用、米国以外用の両方のサブシステムが共に有するブロックであり、2つの入力データに対して所定の処理を施して1つのデータを出力する。フィルタブロック92は、ブロック91からの入力データのうち、所定範囲内の値のデータのみそのまま出力し、それ以外のデータは出力しない機能を有するブロックである。米国では例えば排出ガス中の特定の成分を抑えるために、ブロック91の出力データの上限と下限を法規制により限定しているとする。フィルタブロック92は、この法規制の限定を実現するよう設定される。

#### 【0111】

また、第4実施形態においては、シミュレーションモデル抽出エンジン24は図13のステップ910～960の処理によって実現できるとなっているが、必ずしもこれらの処理のように、不要なサブシステムの除去に切替マトリクス6を利用する必要はない。例えば、図8のステップ810～860の処理によって実現することもできる。また、図5のステップ510～560の処理によっても実現することができる。これらの場合には、図10に示した切替マトリクス6は不要である。

#### 【0112】

また、第1～第4実施形態のコード生成ツール2およびシミュレーションツール7は、その処理の過程において中間コードを生成しているが、必ずしもこのようになっている必要はなく、コード生成ツール2においてはモデル3から直接ソースコードを生成し、シミュレーションツール7においてはモデル3を直接実行

すてもよい。ただし、その際のソースコードからは、モデル3の不要な部分が除かれており、またシミュレーションツール7の実行はモデル3の不要な部分に対応する機能を除いた形で実行される必要がある。

【0113】

なお、本発明の第1～3実施形態においては、パーソナルコンピュータ1がコード生成装置および中間モデル生成装置を構成する。

【0114】

また、第4実施形態においては、パーソナルコンピュータ1がシミュレーション装置および中間モデル生成装置を構成する。

【0115】

また、第1～3実施形態においては、コード生成ツール2がコード生成プログラムおよび中間モデル生成プログラムを構成する。

【0116】

また、第4実施形態においては、シミュレーションツール7がシミュレーションプログラムおよび中間モデル生成プログラムを構成する。

【0117】

また、図5のステップ510、図8のステップ810、図13のステップ910のそれぞれの処理が、モデル取得手段を構成する。

【0118】

また、図5のステップ520、図8のステップ820、図13のステップ920のそれぞれの処理が、選択情報取得手段を構成する。

【0119】

また、図5のステップ530～570、図8のステップ825～870、図13のステップ925～970のそれぞれの処理が、部分指定子および選択情報に基づいて、モデルから部分指定子が指定する部分を取り除いたものからソースコードを生成する除去生成手段を構成する。

【0120】

また、図5のステップ530～560、図8のステップ825～860、図13のステップ925～960のそれぞれの処理が、部分指定子および選択情報に

基づいて、モデルから部分指定子が指定する部分を取り除いたモデルを生成する除去生成手段を構成する。

【0 1 2 1】

また、シミュレーションモデル抽出エンジン 2 4 とシミュレーションエンジン 2 5 が、部分指定子および選択情報に基づいて、モデルから部分指定子が指定する部分を取り除いたモデルを生成する除去生成手段を構成する。

【0 1 2 2】

また、図 1 3 のステップ 9 2 3 が、部分指定子および選択情報との相関関係を示す相関情報を取得する相関情報取得手段を構成する。

【図面の簡単な説明】

【図 1】

本発明の第 1 実施形態に係るコード生成装置としてのパーソナルコンピュータ 1 の構成を示す図である。

【図 2】

コード生成ツール 2 等の構成、作動を概略的に示す図である。

【図 3】

コード生成ツール 2 に入力されるモデル 3 の一例を示す図である。

【図 4】

エンジンの各気筒の状態を加算して出力するブロックの図である。

【図 5】

コード生成ツール 2 の、モデルからソースコードを生成する処理のフローチャートである。

【図 6】

図 3 のモデル 3 から生成される中間モデルの一部の図である。

【図 7】

第 2 実施形態のコード生成ツール 2 に入力されるモデル 3 の一例を示す図である。

【図 8】

第 2 実施形態のコード生成ツール 2 の、モデルからソースコードを生成する処



理のフローチャートである。

【図 9】

図 7 のモデル 3 から生成される中間モデルの一部の図である。

【図 10】

第 3 実施形態のコード生成ツール 2 等の構成、作動を概略的に示す図である。

【図 11】

第 3 実施形態のコード生成ツール 2 に入力されるモデル 3 の一例を示す図である。

【図 12】

切替マトリクス 6 の一例を示す図である。

【図 13】

第 3 実施形態のコード生成ツール 2 の、モデルからソースコードを生成する処理のフローチャートである。

【図 14】

図 11 のモデル 3 から生成される中間モデルの一部の図である。

【図 15】

第 4 実施形態に係るシミュレーションツール 7 の構成、作動を概略的に示す図である。

【図 16】

仕向けによって異なるブロックの構成例を示す図である。

【図 17】

従来の切り替え方法で表現したモデルの一部を示す図である。

【図 18】

図 17 に示した部分に対応するソースコードを示す図である。

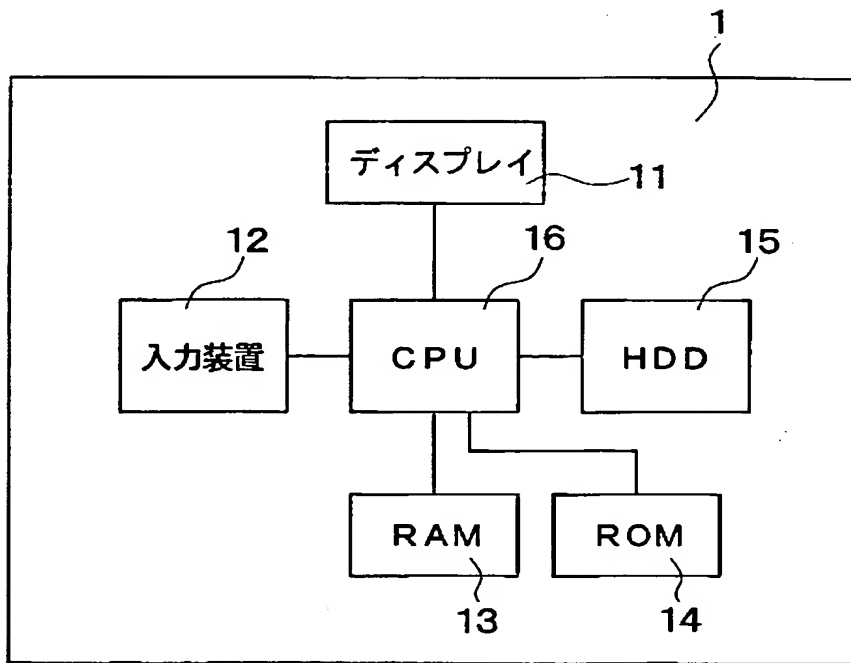
【符号の説明】

1…パーソナルコンピュータ、2…コード生成ツール、3…モデル、  
4…選択情報、5…コンパイラ、リンカ、6…切替マトリクス、  
7…シミュレーションツール、11…ディスプレイ、12…入力装置、  
13…RAM、14…ROM、15…HDD、16…CPU、

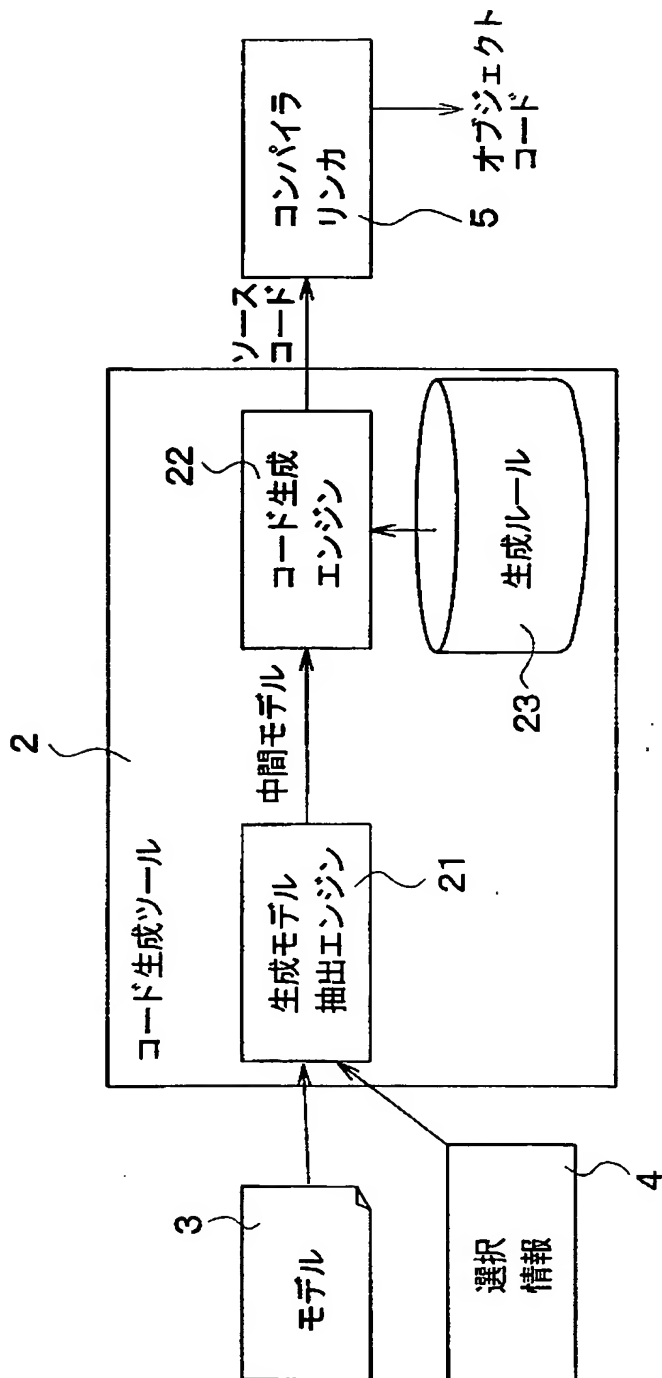
2 1…生成モデル抽出エンジン、2 2…コード生成エンジン、  
2 3…生成ルール、2 4…シミュレーションモデル抽出エンジン、  
2 5…シミュレーションエンジン、3 1…V 6 スタート、  
3 2…V 6 用サブシステム、3 3…V 6 エンド、3 4…V 8 スタート、  
3 5…V 8 用サブシステム、3 6…V 8 エンド、3 7…I 6 スタート、  
3 8…I 6 用サブシステム、3 9…I 6 エンド、5 1…V 6 用サブシステム、  
5 2…V 8 用サブシステム、5 3…I 6 用サブシステム、  
5 4…選択ブロック、5 5…入力端子。

【書類名】 図面

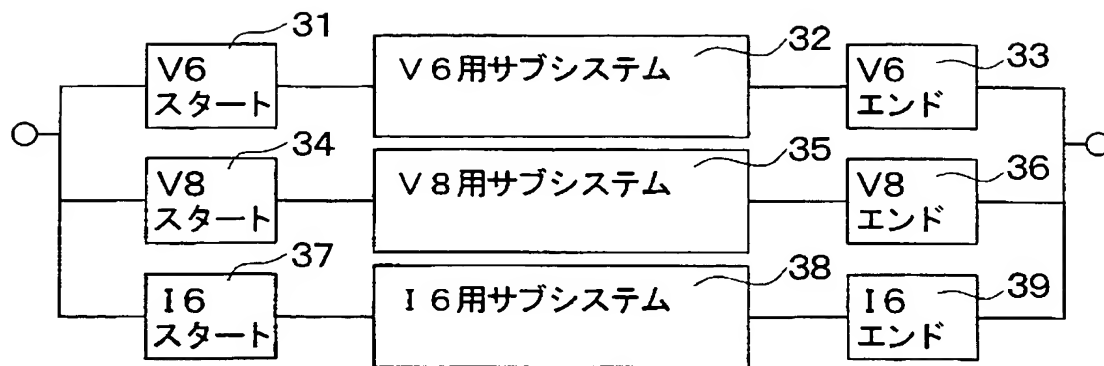
【図 1】



【図 2】

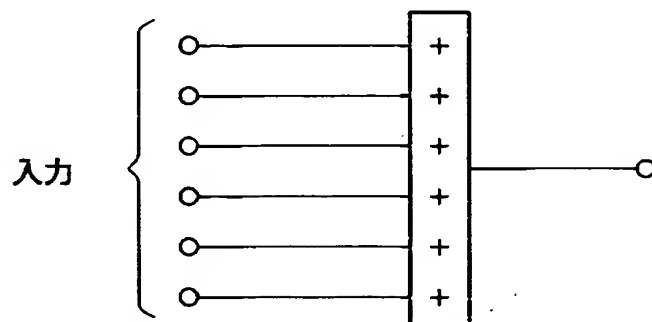


【図 3】

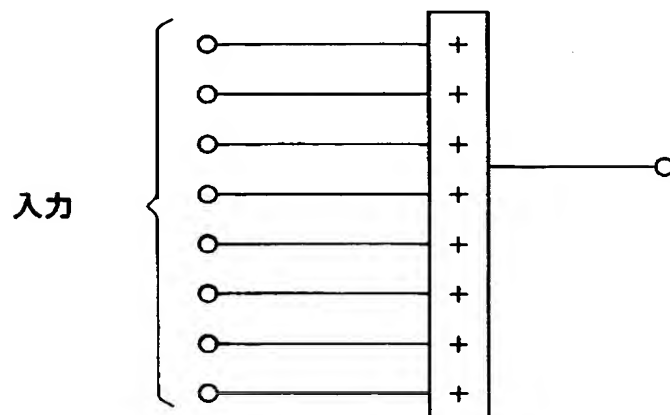


【図 4】

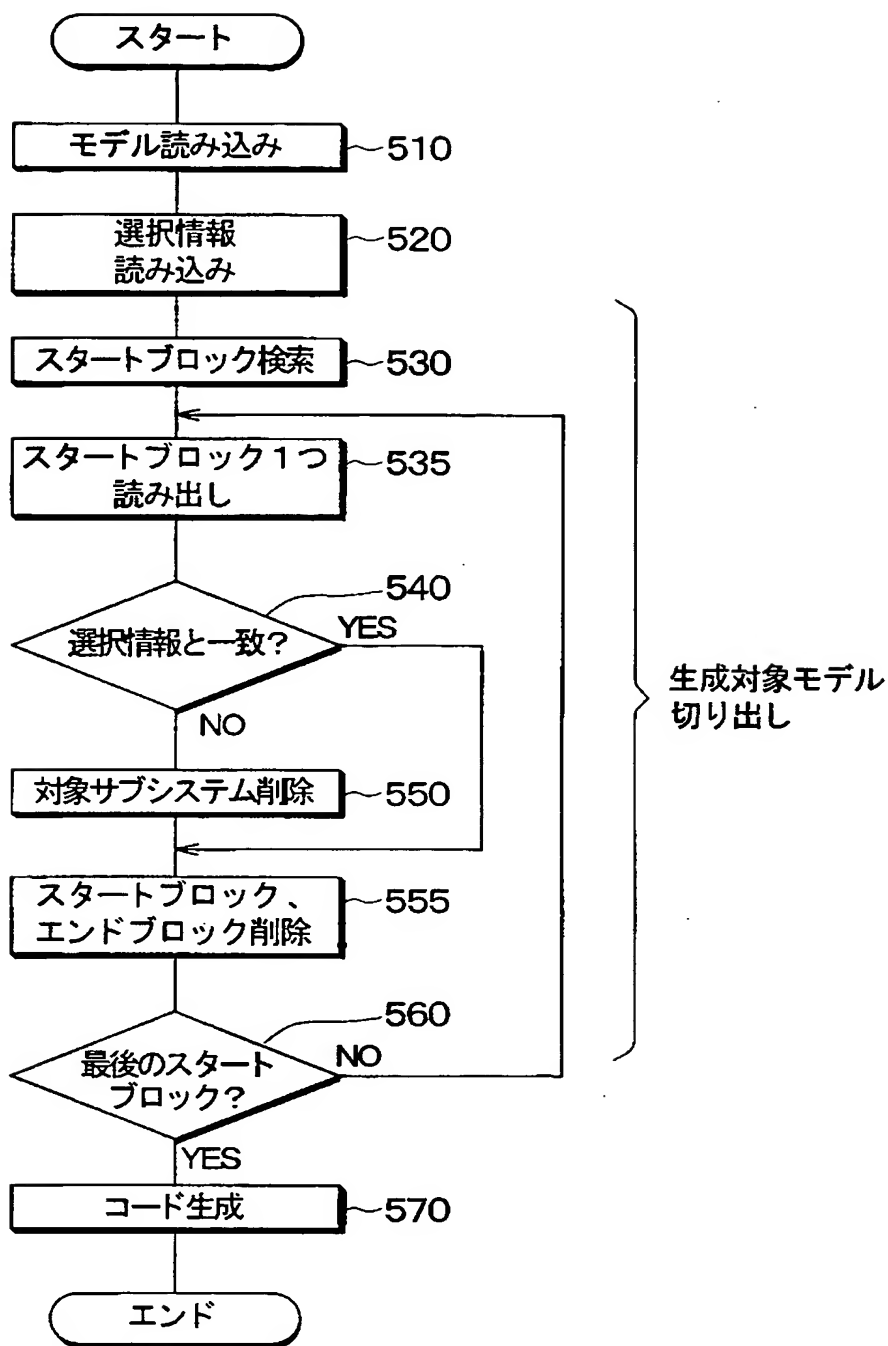
( a ) 6気筒用サブシステム



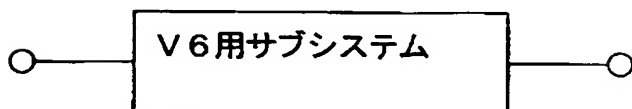
( b ) 8気筒用サブシステム



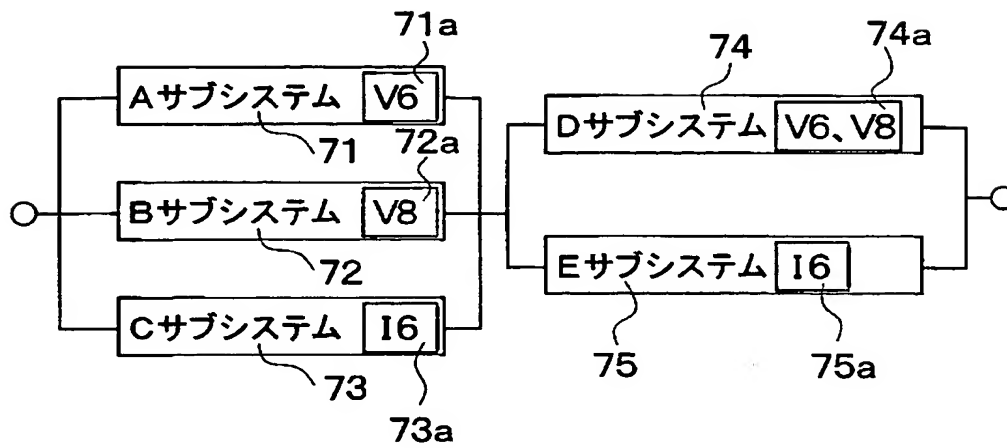
【図 5】



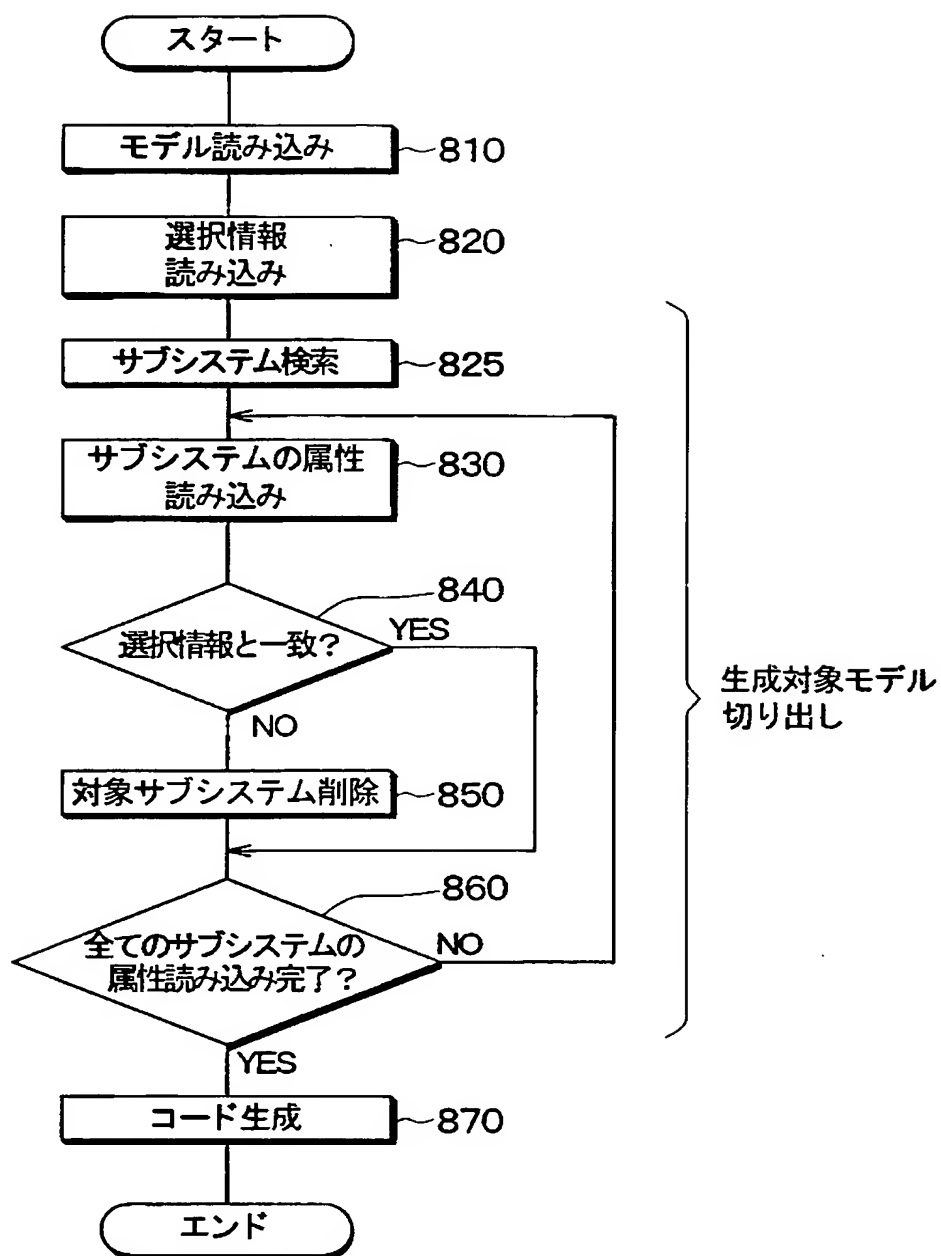
【図 6】



【図 7】



【図 8】

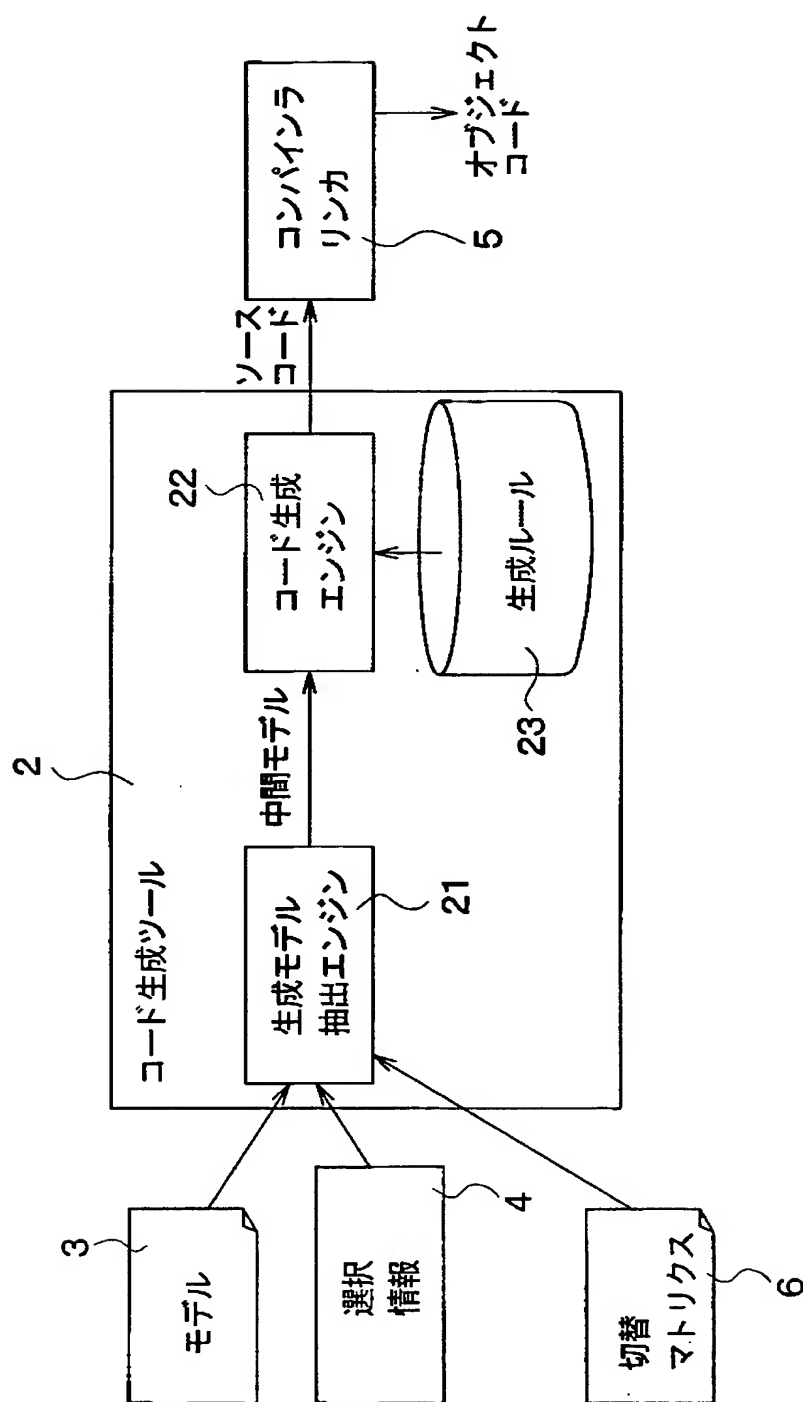


【図 9】

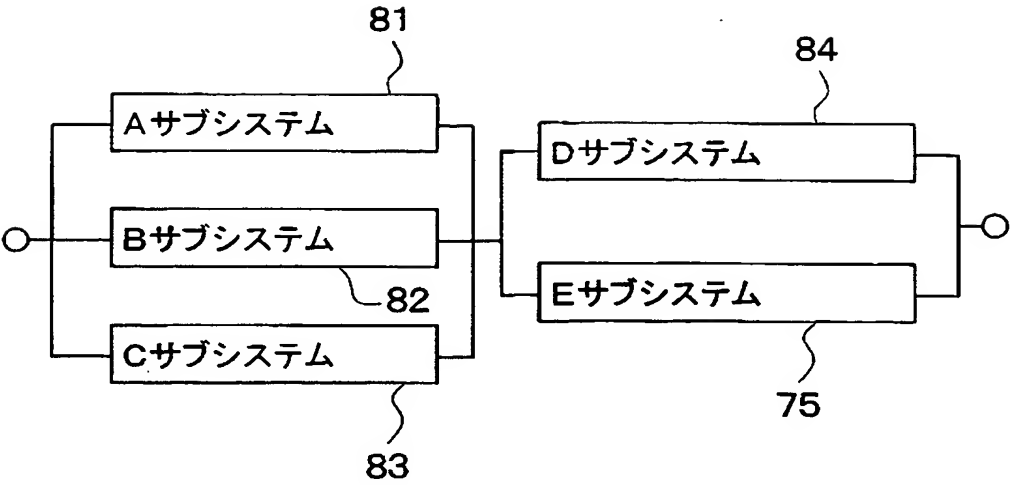




【図10】



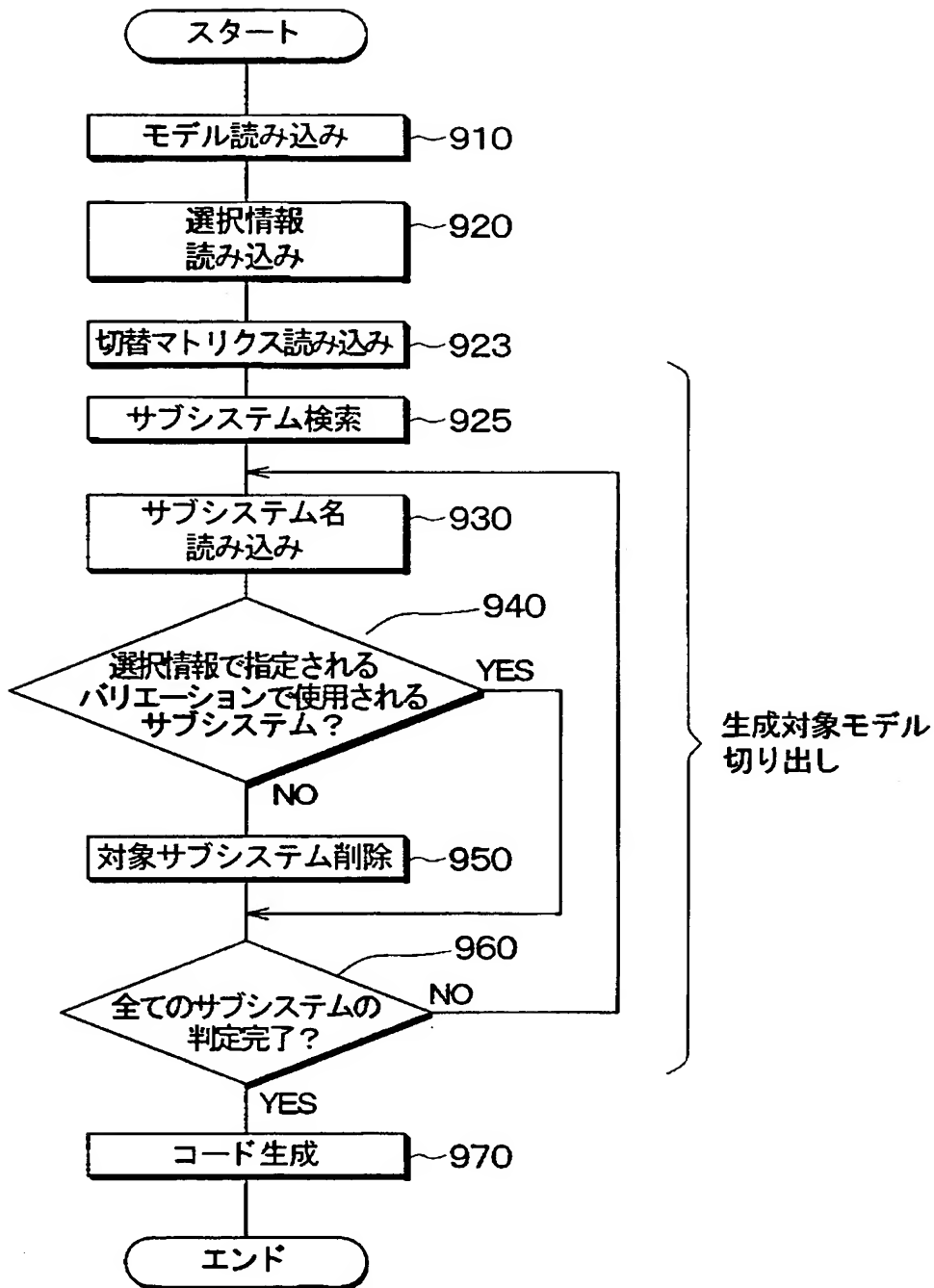
【図 1 1】



【図 1 2】

	V6	V8	I6
A サブシステム	○		
B サブシステム		○	
C サブシステム			○
D サブシステム	○	○	
E サブシステム			○

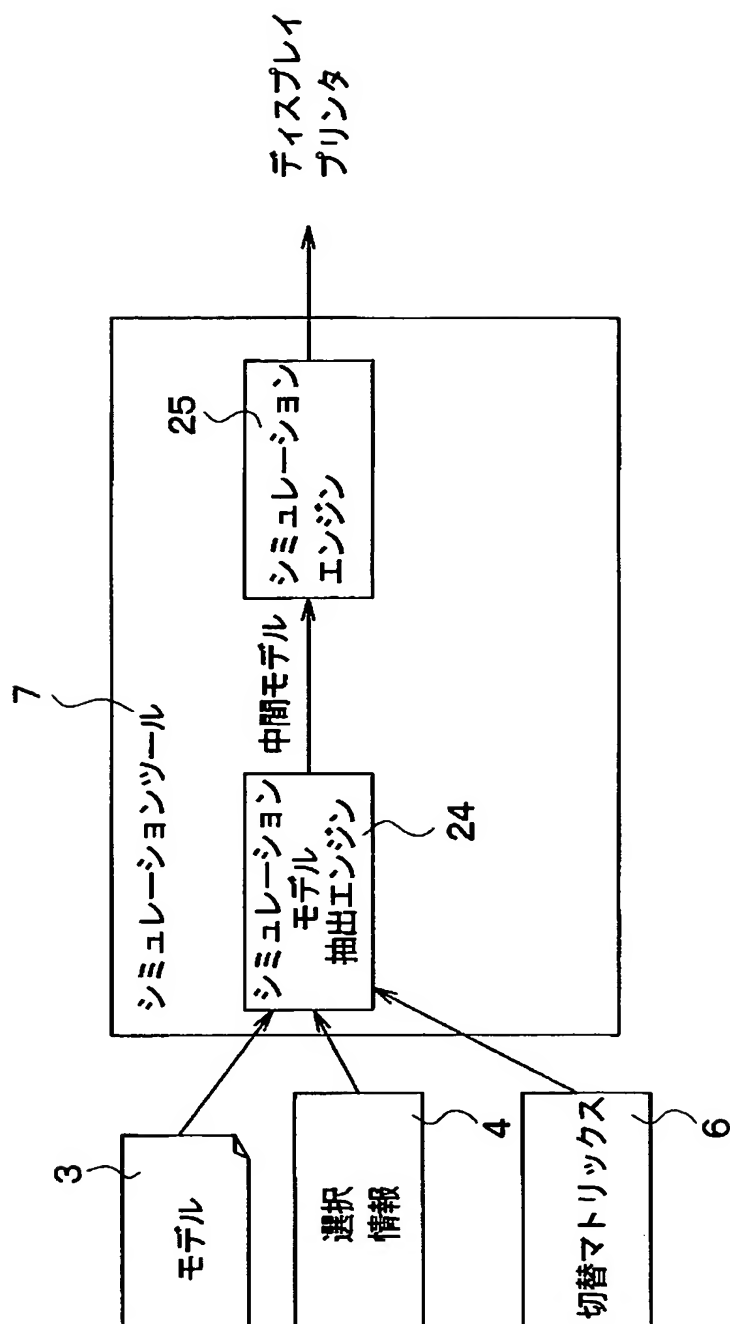
【図 13】



【図 14】

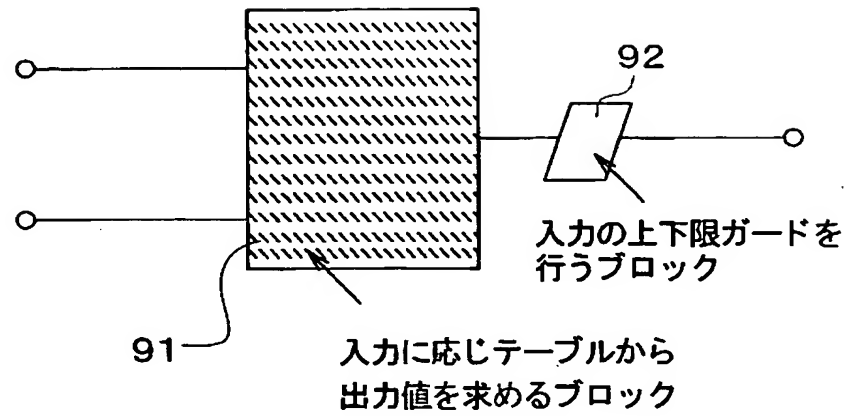


【図 15】

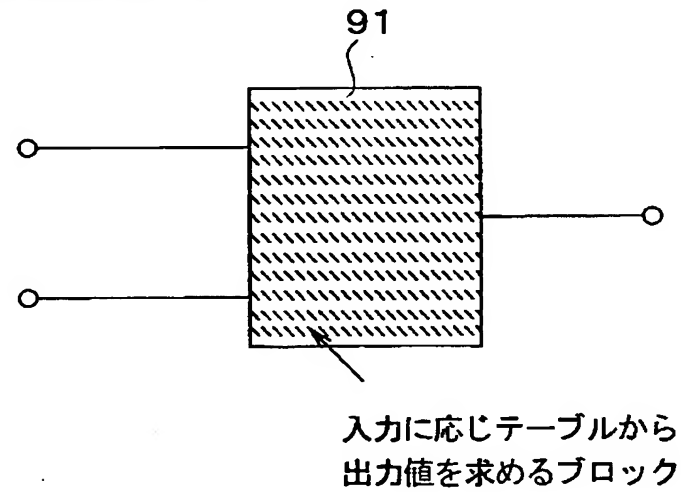


【図 16】

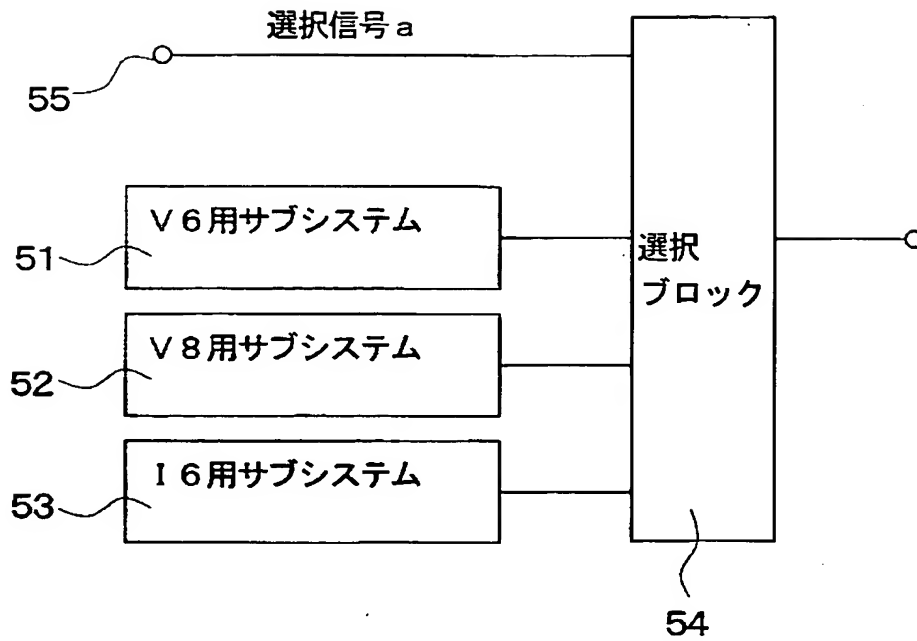
## (a) 米国用サブシステム



## (b) 米国以外用サブシステム



【図17】



【図18】

```
Switch(a) {  
  case1:  
    V6用サブシステム  
    break;  
  case2:  
    V8用サブシステム  
    break;  
  case3:  
    I6用サブシステム  
    break;  
}
```

【書類名】 要約書

【要約】

【課題】 複数のバリエーションに対応するモデルからソースコードが生成される開発環境において、生成されるソースコードから当該モデルの不要な部分に対応するソースコードが除かれているようにすることが可能となる。

【解決手段】 複数のバリエーションに対応するモデルからソースコードを生成するコード生成ツール 2 を備えたパーソナルコンピュータが、HDD から自己の特定の部分を指定する部分指定ブロックを含むモデルを取得し、HDD または入力装置から、部分指定ブロックによって指定された部分の取捨選択に関する選択情報を取得し、この部分指定子および選択情報に基づいて、上記モデルから部分指定子が指定する部分を取り除いたものからソースコードを生成する機能を備える。

【選択図】 図 5

特願 2 0 0 3 - 0 1 7 6 6 8

出 願 人 履 歴 情 報

識別番号

[ 0 0 0 0 0 4 2 6 0 ]

1. 変更年月日  
[変更理由]

1 9 9 6 年 1 0 月 8 日  
名称変更

住 所  
氏 名

愛知県刈谷市昭和町 1 丁目 1 番地  
株式会社デンソー